Copyright versus Patents: the Open Source Software Legal Battle

François Lévêque, Yann Ménière

Review of Economic Research on Copyright Issues, 2007, vol. 4(1), pp. 27-46

COPYRIGHT VERSUS PATENTS: THE OPEN SOURCE SOFTWARE LEGAL BATTLE

FRANÇOIS LÉVÊQUE AND YANN MÉNIÈRE

ABSTRACT. Open Source Software is often viewed as an anti-intellectual property regime. In contrast, we argue how intellectual property law is at the heart of open source model since licenses that organize the innovation and business relationships between developers, distributors and end-users are based on copyright law. The proliferation of software patents can, however be seen as a threat for the development and deployment of open source software. We present the nature of the threat and review a series of initiatives undertaken by the open source community to address them effectively. These initiatives, such as the redesign of licenses and the creation of patent commons, are a testiment to a genuinely creative use of intellectual property law by the open source community, not its undermining.

1. INTRODUCTION

Competition between proprietary and open source software (e.g., Windows versus Linux operating systems for PCs) is often pictured as reflecting an opposition between pro and anti intellectual property law supporters. In fact, the recent EC Commission's attempt to legalize software patents in the EU has been defended by Microsoft and other proprietary software champions while obstructed by the Free Software Foundation and similar Open Source organizations. Such a view is misleading. Open Source Software (hereafter, OSS) relies upon a key body of intellectual property law, namely copyright. Absent copyright law, open source would be unable to license software to developers, distributors and end-users. The great divide between proprietary and open source models lies between patents and copyright. This paper explains why copyright is at the heart of OSS, how software patents may thwart its development and deployment, and why the risk of patent hold-up is decreasing. Section 2 presents the reasons why it is easy to be misled in viewing OSS as an anti-intellectual property model. Section 3 provides a primer on OSS licensing. Section 4 analyses open source licenses from an economic standpoint. Section 5 discusses the conflicting relationships between OSS and patents. Section 6 describes individual and collective initiatives undertaken by open source interested parties to limit patent obstacles. Section 7 concludes.

2. GOOD REASONS TO BE MISLED

There are good reasons for a neutral observer familiar with economics to believe that open source is an anti-IP system. Indeed, economists commonly consider open source software as a form of a public good, which is distributed for free to users and is therefore under-produced by programmers.

Firstly, economic literature is dominated with a single issue:[1] how can developers work for free? Although code cannot be sold by imposing royalties on copies and derived works, millions of lines of open source program have been written since the Free Software Foundation was created in 1985. To explain this puzzle of production without money incentives, economists have analysed individual rewards such as the benefit for a user of improving the software (Von Hippel, 2002), peer recognition, and reputation on the job market for programmers. Such a focus on the absence of financial incentives led to overlooking the role of IP in open source because of the simple but wrong equation 'IP equals Incentives' and, by consequence, the lack of incentives indicates the absence of IP.[2] In fact, the economic analysis of IPRs tends to overemphasize the incentives function of copyright and patent laws. Since Kenneth Arrow (1962), the story goes as follows: creations and inventions are information goods and, by corollary, are non excludable; consequently, creators and innovators cannot recoup the money they invest, and society is confronted with an underproduction of artistic and technical works; IP law makes these goods legally excludable and the free-riding problem that *ex ante* blocks investment for producing information is solved. Other roles of IP law, especially the allocative function in facilitating the diffusion of works through exchange have been neglected. This function is, however, critical.[3] This can be deduced from Coase's (1960) insights according to which granting property rights eases contracting and thus improves allocative efficiency.[4] As we will see in the remaining of the paper, the use of IP in open source is rooted in this exchange function.

Secondly, commons look like public domain. According to a more elaborate story, the entitlement of IP rights law ensures a trade-off between incentives and use. Information goods are both non excludable and non rival. The latter means that consumption does not reduce the quantity available. Therefore, if access to information is restricted to those who pay for it, consumption is inefficiently rationed: certain consumers are excluded (i.e., those who cannot pay the fee) despite the fact that they could otherwise enjoy a private benefit without generating any social cost. In limiting the protection period of patents and copyrights, IP law opens the access to all and reduces welfare losses resulting from charging for non rival goods. In fact, when a patent or a copyright expires, it enters into the public domain and fees can no longer be required for the works that are no longer owned. In other words, protection mitigates the incentives problem while the end of protection mitigates the rationing problem. A common feature between OSS commons and public domain is that both offer access for free. As was said before, no fees limit the number of copies or the possibilities to modify OSS. However, unlike the

[1] A seminal economic paper on OSS albeit bearing the general title 'The Simple Economics of Open Source' focuses on the incentives puzzle. It has been written by two smart and influential economists, J. Tirole and J. Lerner (2002). It has triggered a stream of works on the same issue.

[2] Even Lerner and Tirole (2002) seem not to have avoided the mouse-trap in their first article. Comparing proprietary and open source software schemes they write that owners of proprietary codes cannot "easily allow users to modify their code and customize it without jeopardizing intellectual property rights" (p. 25). It is true that modifications and customisations of code are restricted in licenses of proprietary software. However, this does not mean that IP holders put their IPRs at risk when they alleviate these restrictions. Their copyright or patents will still be valid.

[3] For a general presentation of the economic functions see Lévêque and Ménière (2004).

[4] For a use of the other famous paper on the nature of the firm by Coase (1937) to highlight the organisation of the free software production, see Benkler (2002).

public domain, OSS is owned through copyrights (mostly) or patents (more rarely). OSS commons that regroup pieces of code are not *res nullius*. Their access is free of charge but not unrestricted. Users of OSS under the copyleft license, for instance, must give back their improvements into the commons, which perhaps they would not have done spontaneously. To put it another way, open source uses copyright as a lever to rebalance the trade-off between incentives and use in favour of the latter. It is truly an opposite direction to the historical patterns of increasing IP protection duration, especially in copyright law,[5] but it does not mean the abolishing of IP.

Thirdly, terminology is often confusing. As any scientific and technical community, open source community has forged its own language. OSS was initially baptised 'free software', a name that is easy to associate with the idea of public domain and gratuitousness. 'Copyleft' is another key word used in OSS jargon. It designates the bargain whenever the right to copy and modify the code without payment to the licensor is exchanged with the obligation for the licensee to enrich the software commons with her own contributions. The term wrongly suggests that open source regime rejects and opposes copyright when it actually rejects and opposes the usual use of copyright for an exclusion purpose.

Fourthly, open source organisations and their leaders are prompt to denounce patent system failures. They have played an active role in the EU to block the legalization of software patentability. Hence the misleading syllogism: 'OSS is against patents' 'patent equals IP' thus 'OSS is against IP'. Note that economists are not fully innocent. In economic literature, IP is often used as a synonymous to patent. Intellectual property is a catchy term that encompasses all rights ensuring a trade-off between incentives and use as mentioned above, from those delivered for plant varieties protection to those devoted to database protection.

In a nutshell, in the eyes of economists OSS may look at first glance like an anti-IP system, or at best as copyright unrelated.

3. A primer on open source licenses

About sixty different types of OSS licenses are currently on the shelves. It is out of the scope of this paper to review them.[6] We will only briefly explain what their common features are and describe the most popular ones. In fact, as indicated in Table 1, one license, the so-called General Public License, is widely used and the top 5 licenses represent 86 % of all OSS projects as well as of all developers involved in these projects.

3.1. Open source licenses: definitions.

Open source licenses set the relationship between the copyright holder and everyone who wants to take advantage of the software. They look very unfamiliar relatively to traditional copyright software licenses as well as to IP licenses in general. Commonly, a license restricts the use of the licensee to some field of use, to some geographic areas and to specific products. Moreover, it contains marketing arrangements that specify terms and conditions for licensees to pay royalties to the licensor. For instance, the Microsoft Office suite license indicates that you are not allowed to distribute the software anywhere for

[5]Originally set at 14 years, copyright in the United States has been gradually extended to the current 70 years after the author's death.

[6]For a comprehensive legal analysis of OSS licenses, see Rosen's (2004) reference book; for a brief overview see Kennedy (2001); for the full text of each license see the Open Source Initiative website http://www.opensource.org/licenses/.

any purpose and gives details on the number of copies you are restricted to. An extra payment is required for additional copies. By contrast, OSS licensees grant freedom of use of the licensed software for any purpose and the freedom to make copies of, and to distribute the licensed software without payment of royalties to the licensor. As far as software modifications are concerned, OSS licensees are not restricted. They can create a derivative work, for instance in debugging the software or in improving its customisation, and distribute it. Moreover, the licensee can redistribute her derived work and does not need to pay a royalty to the licensor, not even to report her business. In a nutshell, people are receiving a license that gives them more rights than they have become accustomed to expect under commercial software licenses.

Table 1: The most popular OSI licenses

	Projects		**Developers****	
	Number	**Percentage**	**Number**	**Percentage**
Apache	2,196	2.6	5,234	2.8
BSD	5,968	7.0	15,335	8.2
CDDL	114	0.1	171	0.1
CPL	763	0.9	1890	1.0
Eclipse	241	0.3	506	0.3
GPL	54,842	64.5	112,341	59.9
LGPL	9,429	11.1	23,673	12.6
MIT	1,616	1.9	3,876	2.1
MPL	1,360	1.6	3,937	2.1
Other	8,542	10.0	20,553	11.0
TOTAL	**85,071***	**100.0**	**187,516**	**100.0**

* This figure is a little higher than the number of projects, for we count one project as two in the cases in which the project is registered under two different licenses

** Because nearly two thirds of registered projects in SourceForge involve only one developer, we also estimated the importance of licenses according to the number of developers they gather.

The common features of OSS licenses are enacted in a charter issued in 1998 by the Open Source Initiative. This non-profit organization acts as a certification body. It approves models of OSS licenses, whether they are improved versions of old licenses or genuine new licenses. The charter contains 10 cumulative rules,[7] dealing *inter alia* with non-discrimination against people and fields of use, author's integrity, source code delivery, free redistribution and authorization for software modifications. The fulfilment of all the rules is necessary for a license to be certified by OSI and listed on its website. To make a long story short, the essence of the 10 rules can be summed up in five principles, as follows.

The 5 principles of OSS licenses (source: Rosen, 2004)

(1) *Licensees are free to access and use the source code of OSS.* According to this principle, the licensor of OSS has to make the source code available to licensees at zero price. Moreover, the latter can modify it, in particular to customize the software to his specific needs.

[7]See http://www.opensource.org/docs/definition.php to get the list of, and explanations on the 10 criteria.

(2) *Licensees are free to use OSS for any purpose whatsoever.* Restrictions on use, such as 'for research or non non-commercial purposes only' or 'excluding genetic research' are not allowed.
(3) *Licensees are free to make copies of OSS and to distribute them without payment of royalties to a licensor.* This principle states that a licensee need not pay the licensor for additional copies he makes himself, even if those copies are distributed to others.
(4) *Licenses are free to create derivative works of OSS and to distribute them without payment of royalties to a licensor.* Under this principle, a licensor cannot charge a royalty for the privilege to create and distribute derivative works, or require a licensee to pay a royalty for copies of a derivative works that are distributed, or impose any restrictions on the type or character of those derivative works.
(5) *Licensees are free to combine OSS and other software.* OSS licenses may not impose conditions or restrictions on other software with which the licensed software is merely combined or distributed. This prevents restrictions regarding what other software can be placed on computer storage media or in computer memory.

It is worthwhile making a few general comments on the rules of the OSI charter to cast light on the flexibility that OSS licenses offer.

Firstly, making the source code available is an obligation for the licensor, not for the licensee. As is well known, OSS is synonymous with making the source code available to developers and users. This enables them to correct defects and bugs and customize programs or add features as they deem appropriate. However, they are not required, as licensees, to make available their own source code, that is, the code containing the modifications they introduced into the initial licensed program. Some licenses (e.g., the GPL, see below) impose this requirement, but this is not a necessary condition for a license to be certified by OSI.

Secondly, OSS is completely free for users. As pointed out by Rosen (2004) "OSS can be freely used by anyone, anywhere and for any purpose whatsoever". Whatever the OSS licenses, users can make copies for free and can modify the program according to their needs without any restrictions. The only exception to this grand freedom given to users is that licensees are generally restrained from complaining about software failures[8] and bringing a patent infringement lawsuit (see Section 5).

Thirdly, zero royalty is a key feature of OSS licenses. The source code is available for free. Only payments to cover reproduction costs,[9] if any, are allowed; copies of the licensed program are unlimited and free of charge; copies of derived works are themselves also exempted from royalties to be paid to the owner of the initial work (i.e., the licensor). In proscribing royalties OSS, licenses slightly contrasts with the licensing of IPRs in economic textbooks wherein patent and copyright licenses are structured to maximize licensor's profit. Discrimination, a canonical way to maximize monopolist's surplus, is also proscribed in OSS licenses. Nevertheless, the

[8]Most OSI licenses disclaim liability for damages. No warranties are offered to users about the performance of the licensed program.

[9]Rule 2 of the OSI charter on source code states that "Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost."

reader must not conclude that OSS licensors and licensees cannot make money in selling code. OSS licenses are not exclusive. Copyright owners can also license their code with another license, providing for instance, a warranty to customers. Such a multiple licensing is exemplified at the end of this section. Moreover, licensees may sometimes also license their derivative works under non-OSS licenses and therefore can ask royalties to their licensees.

OSS licenses differ in several respects. The key one is whether the license explicitly restricts the choice of licensees in licensing their derivative works. The GPL is the most influential license containing such a restraint. If a licensee improves the code of a GPL-licensed piece of software, he should also use the GPL license to distribute her new piece of software. As a consequence, he will have to make the improved code available and to permit copies and redistribution at zero royalty. The GPL license was first published in 1989. It was written by Richard Stallman, the founding father of the Free Software Foundation, and Eben Moglen, from the Columbia School of Law. Their basic idea in imposing such a restriction, the so-called "copyleft", was to ensure software access to developers and to prevent software from being captured by proprietary software interests. Access would be ensured since licensees cannot selfishly remove their improvements from the collective resource available to developers. Capture would be avoided, since royalties cannot be claimed. A major factor in the spread of popularity of the GPL in the developers and users communities is that is has been adopted by Linus Tordvals for the development of Linux. The Linux operating system and programs associated with Linux are mostly licensed under GPL licenses.

During the eighties, Bill Joy from the University of California at Berkeley forged another type of license that has become very influential and widespread too, the Berkeley Software Distribution (hereafter, BSD) license. Unlike the GPL, the BSD license does not require that the derivative works also be subject to the same terms as the initial GPL license. BSD licensees can improve the software and distribute their derived works without any obligation regarding source code disclosure and royalty payments. They can integrate the software provided by the licensor in any proprietary software of their own and may adopt proprietary licenses. BSD licenses can now be found in commercial software such as Windows NT and the Apple's operating system, OS X.

GPL and BSD licenses have served as models for many other OSS licenses. In fact, they gave birth to two families of licenses, the so-called academic licenses for BSD-type licenses and the so-called copyleft (or reciprocal) licenses for GPL-type licenses. Generally speaking, they are more precise and legally sound, for they benefit from discussions on first OSS licenses and on their loopholes and failures. The MIT license and the Common Public License (CPL, hereafter), which have been written by professional lawyers, are good examples of more modern academic and reciprocal licenses, respectively. Indeed, both make explicit the rights that are granted to licensees in terms of copyright law. CPL, for instance, states in Section 2(a) that the licensor grants licensees "a non exclusive, worldwide, royalty free copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, distribute and sublicense" the considered software. More recent licenses are also clearer with respect to the use of the name of the licensor and its software. The Apache license, for instance, specifically excludes the granting of the

trademark to licensees. It requires an acknowledgment in end-user documentation and it proscribes the use of the name 'Apache' to promote derived works.

Copyleft licenses are not uniform regarding the scope of the reciprocity obligation. The GPL obligation is the broadest. It states in section 2(b) that "you must cause any work that you distribute or publish, that in whole or in part contains or is derived from the program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this license". This statement has raised concern about the possibility to combine GPL-licensed software with proprietary software in collective works and larger works that include works derived from GPL-licensed initial work. As a response, the Free Software Foundation developed the Lesser General Public License (LGPL, hereafter) that explicitly allows such a combination to be made through the use of programming libraries. Such a use does not infect the proprietary software in requiring it to be subject to the provision of GPL. The Mozilla Public License (MPL, hereafter) is also much less restrictive than GPL. Its reciprocity obligation concerns only the files containing the derivative works and not the derivative works more broadly. MPL-licensed software can be used as building blocks to create larger works, for the latter may be open or proprietary. As pointed out by Rosen (2004), MPL acts like an academic license for larger works but the individual building blocks are licensed with reciprocity obligations.

3.2. **Open source licenses and copyright law.** To conclude this general presentation, we would like to emphasize on the encrustation of OSS licenses in copyright law. Generally speaking, an OSS license is a copyright license; the licensor owns the copyright of the software; he grants a generous license but this must not be viewed as she surrenders her copyright. As already said, OSS must not be confused with public domain. Take the GPL, for instance. It is a bare copyright license and relies entirely upon copyright law for its enforcement. Section 5 states that "nothing else [than the license] grants you the permission to modify or distribute the program or its derivative works. These actions are prohibited by [Copyright] law". The MIT license is another example. As already mentioned above, the licensor explicitly grants to the licensee all his exclusive copyright rights to copy, modify, distribute, publicly perform and publicly display as listed in the US copyright law. Copyright ownership being a necessary condition to license OSS, only the copyright holder can decide to adopt and issue other licenses.

Because initial licenses contain loopholes, new versions including clearer definitions and additional provisions are required. For instance, a new version of the Apache license was released in 2004, in order *inter alia* to include a patent defense clause (see section 5) and to clarify the submission of contributions. As another example, a revision of GPL has been intensively debated for the past two years. The final text of the new GPL template is likely to be released at the end of 2006. Some OSS license migration can also take place. The open source ERP software OFBiz, for instance, shifted from MIT to Apache 2.0 to provide a better legal security to end-users. Of course, only copyright holders can decide OSS relicensing. Such changes are thus difficult when open source projects are in the hands of many copyright holders since each contributor has a say. To avoid such a burden, an alternative consists in centralizing the rights at the project level through assignments or delegation. The Free Software Foundation, for instance, asks for copyright

assignments from authors of code incorporated in FSF projects. The Apache Foundation Software requires its contributors to submit a signed contributors' license agreement that provides the foundation with flexibility for relicensing. Since only copyright holders can defend themselves against copyright infringement, note that such centralization enables also a better enforcement of OSS licenses.

Only copyright holders too can propose different licenses for the same software, an OSS license and a proprietary license. MySQL, for instance, provides its database software both under the GPL for OSS developers and distributors and a commercial license for OEMs. Ghostscript, a Postscript interpreter, proposes a commercial license for the most recent version of the software and a GPL for older versions. These multiple licensing strategies offer growing business opportunities for OSS deployment.[10]

4. Basic economic analysis of OSS licenses

4.1. **Incentives and use.** The requirements to make the source code available and not to ask for royalties maximize the use and diffusion of software innovation and minimize the cost of R&D duplication. Let us explain this statement. As already mentioned, IPRs operate a trade-off between incentives and use. Since OSS licenses of initial works cannot impose royalties on copying and modifying the software, the cost of access for users and developers equals the marginal cost of reproducing and communicating the code and the related documents. This cost is close to zero, for most of this information can be downloaded from the Internet. In other terms, the OSI licenses follow the economic prescription that information as a non-rival good must be priced at its marginal cost of communication. In principle, the reverse of the medal is a loss in incentives – and thus in innovation –, for licensors cannot recoup the costs of their efforts through royalties. In fact, they enjoy other rewards. There is a long list of speculative or empirically observed motives in literature that explain the benefits developers get outside of royalties. It suffices here to say that in so far as developers choose an OSS license, they expect to find one way or another to compensate their cost. The reason is that an OSI certified license is not their single option. When royalties are necessary for some developers to recoup costs or for some software to be developed, proprietary licenses can be used as an alternative. We may therefore infer that the non-rivalry prescription is not achieved at the detriment of incentives. Whether OSS licenses decrease investment in developing software because of the monetary incentive loss they entail is only an issue for the GPL and other reciprocal licenses. For these licenses, the licensee has no other choice than distributing his derived work without royalties for copies and subsequent modifications. If he is not sensitive to non-monetary rewards of OSS, his single alternative option is not to invest in modifying the GPL-licensed software. One cannot therefore exclude that some welfare enhancing innovations will not be made. GPL adversaries often claim that copyleft licenses hinder software innovation, especially those suited to end-users needs because they discourage participation from commercial software companies.

The free access to source code maximises the diffusion of the knowledge contained in the protected software. It plays the same role as patent disclosure in enabling R&D spillovers. It reduces duplication costs, an important saving for society because reinventing the wheel is a waste of resources. Interestingly, the decrease

[10]For a presentation of business model-OSS license nexus see Valimaki and Oksanen (2002).

in duplication cost is amplified by the absence of royalties in licenses. Generally speaking, licenses reduce duplication cost since the choice for a licensee is either to license or to work around the protected innovation. Of course, he opts for a license for which the cost of the license is lower than the R&D costs of working around. In so far as OSS licenses are royalty free, the cost of the license is zero and thus R&D is never duplicated. Again, copyleft licenses are a special case. The choice for a GPL license is costly for the licensee because he may lose the opportunity to make money in closing the code and claiming royalties for copies. He will have to choose to duplicate the GPL-licensed code in rewriting it or to accept the GPL license with its reciprocal obligations.

In principle, the reverse of the medal is a loss in incentives - and thus in innovation - for licensors cannot recoup the costs of their efforts through royalties. In fact, they enjoy other rewards. There is a long list of speculative or empirically observed motives in literature that explain the benefits developers get outside royalties. It suffices here to say that in so far as developers choose an OSS license, they expect to find a way or another to compensate their cost. The reason is that an OSI certified license is not their single option. When royalties are necessary for some developers to recoup costs or for some software to be developed, proprietary licenses can be used as an alternative. We may therefore infer that the non-rivalry prescription is not achieved at the detriment of incentives. Whether OSS licenses decrease investment in developing software because of the monetary incentive loss they entail is only an issue for the GPL and other reciprocal licenses. For these licenses, the licensee has no other choice than distributing his derived work without royalties for copies and subsequent modifications. If he is not sensitive to non-monetary rewards of OSS, his single alternative option is not to invest in modifying the GPL-licensed software. One cannot therefore exclude that some welfare enhancing innovations will not be made. GPL adversaries often claim that copyleft licenses hinder software innovation, especially those suited to end-users needs because they discourage participation from commercial software companies.

4.2. **Open source commons.** Another key difference lies in the building and the size of the commons.[11] GPL creates a vast commons for it obliges developers to give back their modified works to the commons. The size of the commons grows with the evolution of the projects. On the contrary, under an academic license, developers cannot be refrained from making their modifications proprietary. In free-ridding, they will benefit from a collective effort without contributing to it. It seems (Rosen, 2004), however, that despite the absence of reciprocal obligation, many contributors to software projects under BSD or similar licenses pour their modified code into the commons. Putting it another way, they see more value to themselves in giving software away than in keeping it private. Everything being equal, the commons of GPL source code would be larger than the commons of code under academic licenses. Of course, other factors enter into account. Fershtman and Gandal (2005)

[11]From an economic point of view, this term is misleading for commons are non excludable and rival goods. It is merely because they are rival that commons may tragically become depleted like meadows by overgrazing when property rights are not entitled. By contrast, source code is excludable (e.g., through secret, technical features and IP law) and non-rival. OSS licenses make the code non excludable but the code remains non rival. Anyway, we will also use the term commons here for it is widely used in OS literature to emphasise that access and use of OSS are free.

have demonstrated, for instance, that the output per contributor (i.e., the number of lines of code) is lower for projects under copyleft licenses than for projects under academic licenses. As an explanation, they suggest that developers contribute up to the threshold level necessary to have their name listed on copyleft-licensed software whereas they have financial incentives to go beyond with academic licenses. Larger individual contributions for academic licenses may therefore compensate the smaller size of academic commons resulting from the absence of reciprocal obligation.

The size of the commons also depends on the attractiveness of the different licenses for project leaders and contributors. The majority of project leaders seem to prefer copyleft licenses, for these represent a large majority of licenses they choose (cf. table 1). However, as pointed out by Lerner and Tirole (2005a and 2005b), in some cases leaders may prefer an academic license but opt for a copyleft license to get the participation of more developers in their projects if the latter would prefer this type of license. In their analysis of 38.000 projects posted on SourceForge, they found that:

- projects with academic licenses are larger;
- the choice of licenses differs depending on whether projects are oriented towards end-users (e.g., game software) or applications for software developers (reciprocal licenses are more common in the first case whereas academic licenses are more frequent in the second);
- copyleft licenses are less common for more mature projects.

Finally, project leaders may adopt this or that license just because others have done so. Learning costs to know the subtle differences between all OSS licenses are high. Old licenses, even if they contain loopholes and ambiguities, enjoy the advantage of historical standards. This may explain why, for instance, the MIT license, albeit more precise and legally sound, has not supplanted the BSD license (cf. Figure 1).

Finally, it is important to notice that there is not a single OSS commons. The reason is that there are several licenses some of which are incompatible. The current number of licenses listed on the OSI site is 59. Getting a clear picture of their similarities and differences needs time and effort.[12] Of course, the multiplication of OSS licenses[13] has improved the choice for project leaders and software companies to find the license that best suits their needs. It seems, however, that this advantage does not offset the major drawback of license multiplication: the partitioning of OS commons. The goal of open source has been to make it easier to reuse code written by someone else. Incompatible licenses limit reuse and may result in a tragedy of the anticommons. It is not possible for instance to add a GPL-licensed contribution to an Apache-licensed derivative work. Instead of having a single commons wherein a developer collects different pieces of software to derive a new work, he must restrict his choice of software to, say, Commons A, B and D and cannot graze in commons C. Generally speaking, academic licenses are more compatible than copyleft licenses. A GPL-licensed code can be combined with BSD-licensed code and the derived work distributed under GPL. The reverse is impossible: a BSD-licensed project can use

[12]Lawrence Rosen's (2004) complete and thorough book on open source licensing contains 314 pages. The full text of all approved licenses available on the OSI website amounts to more than a thousand pages.

[13]In October 1999, the OSI website listed about a dozen licenses. The number increased to 35 in Fall 2002 (Peters, 2006).

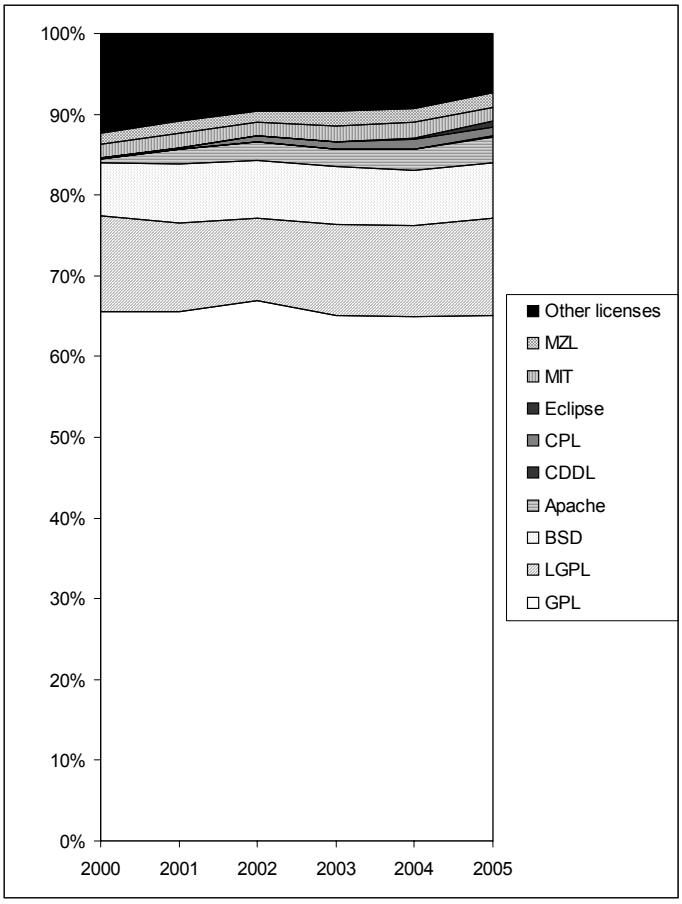

FIGURE 1. The evolution of license choice (Each year new OSS projects are listed on SourceForge and project leaders indicate the license they choose. The figure shows that the historical licenses, namely, GPL and BSD remain by far the most popular.

and redistribute GPL-licensed derivative work only if the project is relicensed into GPL. Generally speaking, the game is not worth the candle since, as was said before, relicensing requires the permission of all copyright owners. An alternative may consist in accepting the legal risk. Incompatibility of, say, MPL or Apache with GPL or LGPL is not iron-cast. In spite of claims by the Free Software Foundation to the contrary, whether a developer could be successfully sued on grounds related to these perceived license incompatibilities is unclear.

The issue of license multiplication has been tackled by OSI, and they created a license proliferation committee at the end of 2005. OSI seeks to reduce the increase in the number of licenses and to enlarge the use of the most popular ones. The first draft report of the committee was issued in July 2006. It has selected 9 licenses, including *inter alia* BSD, MIT, GPL, LGPL, Apache, and MPL as popular and widely used or with strong communities. It encourages licensors to select their

license from this group. OSI also tries to discourage registration of new licenses. In recent guidelines, it has recommended that licenses must not be duplicative. In fact, signalling has been one reason for firms to register their own OSS license. These vanity licenses are one cause of license inflation. Lucent, IBM, Sun, Nokia, RealNetworks, Apple, and Ricoh, for example, all possess their own OSI license. They are either redundant with more popular licenses or cannot be reused for they are specific to products or projects. Some authors of licenses such as Intel and Jabber have put down their ego in withdrawing their OSI certified license. Mozilla foundation's relicensing is another initiative to cope with the tragedy of the anticommons that license proliferation entails. The Mozilla source code is now open under a triple license GPL, LGPL and MPL. This means that developers can combine Mozilla code with LGPL and GPL code and redistribute their derived work in using one of the three licenses.

5. Patent obstacles for OSS development and deployment

As shown in the previous section, OSS development and deployment rely upon copyright. By contrast, most OSS supporters are fierce opponents to software patents. Software patents are generally viewed as raising serious obstacles for the growing use and development of OSS. We evaluate this assertion in this part. We recall first the problems raised by patents in the proprietary software industry, and discuss as a second step whether they also concern open source software communities.

5.1. **Software patent, hold up and defensive patenting.**

The software technology is such that one software program may embody a large number of patented elements. Such patents are broader than copyright. Therefore they encompass a wider range of software applications and they are more difficult to circumvent by rewriting the code. Moreover, it may be difficult to identify all those patents in advance, especially because the software industry has primarily developed in the realm of copyright, so that information on prior art is still difficult to collect. In this context, the risk of patent "hold up" is important.

The problem of hold up is frequent in ICT industries where products usually encompass a large number of patents[14] some of which may be difficult to identify at early stages of innovation[15] (FTC, 2003). It is more acute when the patented element represents only a small fraction of the profits generated by the technology for the patent owner can then leverage the threat of an injunction to impose a fee that largely exceed the intrinsic value of the patented element (Lemley and Shapiro, 2006).

Empirical evidence shows that large capital intensive companies are the main cause and the main victims of the threat of patent hold up (Hall and Ziedonis, 2001). Their large patent portfolios confer them more opportunities to initiate profitable patent litigation. Yet because of their wealth, they are also the most

[14]More than "90,000 patents generally related to microprocessors are held by more than 10,000 parties", while "approximately 420,000 semiconductor and system patents [are] held by more than 40,000 parties." (FTC, 2003).

[15]Even when infringed patents are identified early enough, it is often difficult to assess whether these patents are really valid, for many software patents are in fact fragile in this respect (Burk & Lemley, 2005). Such uncertainty on the legal validity of patents makes it more difficult for developers to strike licensing deals in order to clear the way for innovation.

valuable targets for opportunistic patent litigation. They reply to such threats by filing more and more patents which they can use as counter threats against each other, thereby thickening the patent thicket (Hall and Ziedonis, 2001; FTC, 2003).[16] The balance of patent threats eventually enables them to negotiate cross-licensing agreements whereby they grant each other freedom to operate on their respective patent portfolios.[17] Large firms are however still threatened by small firms – so called "patent trolls" – that specialize in patent litigation without any productive activity. Indeed the counter-threat of a reciprocal patent litigation does not operate on that kind of opportunistic predators.

By contrast, empirical evidence shows that small software firms tend to use patents merely as a protection mechanism to preserve incentives to innovate, or as a signalling mechanism to attract venture capital (Hall and MacGarvy, 2006). Although they do not represent profitable litigation targets for purely opportunistic patent holders such as trolls, they remain exposed to litigation initiated by competitors. In this case small firms can neither leverage a large patent portfolio to counter the threat of litigation, nor afford the high cost of a patent litigation.[18] Therefore they tend to specialize in niche markets to minimize the risk of patent litigation, as it has been observed in the semi-conductor industry (Hall and Ziedonis, 2001).

5.2. **The patent threat on open source software.** The threat of patent hold up also concerns open source software. Patent thickets tend to surround open source software, all the more so as large software and hardware firms get increasingly involved in opens source projects and thus tend to extend their patent portfolios in that direction. A study by Open Source Risk Management (2004) found for instance that 283 patents could potentially be used to support claims of infringement against the Linux kernel.[19] In fact, the key question is whether the probability of infringement is higher for OSS than for proprietary software. It is commonly said

[16]Such defensive patenting strategies are typical of the semi-conductor and computer hardware industries (Hall & Ziedonis, 2001; FTC, 2003). They also exist in the software industry, in part because of the influence of hardware firms in this sector. In the late 1990's, hardware firms accounted for more than 50% of U.S. software patents, and this share was increasing (Graham & Mowery, 2003).

[17]In the FTC report on Patent and Innovation (2003), these strategies are labelled "MAD". The acronym stands for Mutual Assured Destruction – the likely outcome of a full litigation between two large firms asserting their patent portfolios against each other.

[18]Note that the risk of patent dispute depends on your defences as perceived by your adversaries. In this respect, the asymmetry between small- and large-sized enterprises is high. Resolving patent disputes in the courts costs several millions of US dollars. This amount is too high to be afforded by individual developers and small firms. The model-railroad control software dispute gives a good illustration. Bob Jacobsen developed a model-railroad control software and gives his work away with full source code. He is faced with an invoice for over US $ 200.000 from the company KAM. This company filed a patent making a broad claim covering the transmission of model-railroad command between multiple devices in 2002. According to B. Perens (2006), the patent is likely to be invalid for lack of novelty for the technology goes back to the MIT model railroad club in the 1960s. However, Bob Jacobsen could easily bankrupt in defending himself.

[19]These patents have not been reviewed by the courts; it is unknown whether they are valid and/or they are infringed. As pointed out by Lemley and Shapiro (2005) a patent is only a probabilistic property right. Litigation and court ruling are necessary to assert if the patent holds. In average, when a patent is challenged in the US court system, the court finds it invalid about 40% percent of the time. For this figure concerns patents in all technical fields, not specifically software, one cannot apply it to the Linux case. It will be very unlikely, however, that the 283 patents would all be invalidated.

that most software contain portions of code that potentially infringe patents and that it is difficult today to write a significant program without using a principle covered by a software patent. The OSRM study emphasizes that 'if we were to study any other operating system, including Windows, we'd come up with at least as many patents, if not more' (McLaughlin, 2005). It is just a guess. Empirical evidence lacks to know how OSS and proprietary software differ in this respect. Considering that OSS is issued from smaller and less commercial organisations, one may infer that the OSS probability of patent infringement is higher for OSS than for proprietary software. OSS organisations have less needs and less money to check whether code potentially infringes patents. However, it is commonly said that large commercial companies do not check code for patent infringement.[20]

An important difference between proprietary and open source software is that in the latter potential infringers form a community. Patent owners must therefore decide which member of the community they will sue. A firm that produces and commercializes its proprietary software is an obvious target for an infringement suit. By contrast, open source software is produced by atomistic communities of users-producers who do not commercialize their production. The problem for an opportunistic patent owner is thus to determine how to derive a profit from the infringement of his patent by an OSS. Against this background, hold up essentially makes sense if the OSS is combined with complementary products, processes or services that generate profits for the OSS user. This implies that the threat of hold up does not really concern "pure" OSS users, but rather corporate OSS users. The catastrophe scenario imagined by Bruce Perens, the creator of the OSI definition, illustrates this.[21] Suppose that the Linux operating system infringes in a substantial part patents held by some major proprietary software firm. That could result in liability for all of the many firms using Linux jointly with commercial goods from cell-phones companies to commercial banks. They would have to pay a fee or to stop to use Linux, a costly solution too for their costs to switch to another system are high.

Amongst open source software users, large companies with "deep pockets" remain the best targets for patent hold up. The SCO v. IBM case illustrates this, although it involves copyright and not patents.[22] Large companies using OSS can

[20]Firstly, such a screening is very lengthy and costly. Secondly, it requires reading patents and in looking at patents companies take the risk to pay for treble damages in case of litigation. Moreover, large commercial software companies have invested in defensive patents: they can use their own patents as a bargaining ship in case of dispute (i.e., when both parties have patent portfolios each could sue the other in a similar way, so neither one does).

[21]"A coordinated patent attack by a few companies, or even one large company, could completely destroy Open Source in the United States and cripple it in other nations" The Monster Arrives: Software Patent Lawsuits Against Open Source Developers, June 30 2006, available at http://technocrat.net/d/2006/6/30/5032.

[22]Ironically, up to now, the main case of litigation against Linux, SCO vs IBM, deals with a copyright and trade secret dispute not with a patent infringement. The initial step of the case is very close to the story told in the introduction. On March 6, 2003, the SCO group filed a $1 billion lawsuit against IBM claiming IBM has, without authorization, contributed SCO's IP to the codebase of the open source, Unix-like Linux operating system . In May 2003, SCO group sent a letter to members of large US firms warning them of the possibility of liability if they use Linux. Another series of letters was sent in December 2003 alleging copyright infringement related to 65 files in the Linux code tree. OSS supporters have characterized SCO's actions as an attempt to dissuade developers, distributors and end-users to work with Linux OS in creating fear, uncertainty and doubt (FUD) in their mind. The lawsuit is still in the discovery phase. It

still prevent a part of the threats by leveraging their own portfolio against aggressive patent holders. As a matter of fact, a growing number of pure OSS companies[23] pragmatically file patents. Red Hat for instance has created its own portfolio of defensive patents.[24] However such individual defensive strategies increase the legal threat on the whole open source community, by fuelling even more the growth of patent thickets. The development of small OSS companies can also be jeopardized by patent litigation. Since small companies do not have "deep pockets", they are not really threatened by patent trolls. However, they may still be sued by larger competitors that wish to push them out of the market. Following the example of small proprietary software companies, they may be obliged to stay in more secure technology niches instead of commercializing more generic software to consumers.

Beyond exposing individual companies to litigation threats, software patents create more generally a climate of legal uncertainty that is detrimental to the OSS community as a whole. Individual exposures of OSS users to legal risks are not independent: if an OSS user is held infringing a patent, then all other users and developers of the same software are *de facto* infringers. This collective exposure to patent hold up reinforces the incentives for opportunistic patent owners to attack OSS users, because it generates economies of scale in litigation. (Farrell and Shapiro, 2006). This is especially the case when OSS are used by a large number of firms, as with Linux. The legal uncertainty due to patents (but not only to patents), is sometimes called FUD (which stands for "fear uncertainty and doubt"). It is one the main obstacles to the diffusion of open source software amongst companies. As we shall see, it also creates incentives for OSS communities in general, and OSS companies in particular, to organize collective defences against patent threats.

6. OSS defences against patents

In the future, *SCO v. IBM* might appear as beneficial to OSS. It has initiated a quick and large learning process on IP law amongst the OSS developers and has raised their awareness on the importance of legal issues. It has triggered a series of individual and collective initiatives to reduce the patent threat to the development and deployment of OSS. Indeed the protection against legal risks is to a large extent a *collective good* for communities that are already dedicated to the collective

is therefore far to be solved. However, it seems likely the outcome would differ from the story sketched out in the introduction of the section. SCO has failed to identify the version or the line numbers of the Unix code it claimed had been inappropriately transfer in Linux code. The hypothesis in our story about a substantial part of Linux held by the plaintiff, namely SCO, does not hold here.

[23] In reality these companies are not "pure" since they sell services that complement the OSS.

[24] Red Hat explains its patent policy as follows: 'Red Hat has consistently taken the position that software patents generally impede innovation in software development and that software patents are inconsistent with open source/free software [...] At the same time, we are forced to live in the world as it is, and that world currently permits software patents. A relatively small number of very large companies have amassed large numbers of software patents. We believe such massive software patent portfolios are ripe for misuse because of the questionable nature of many software patents generally and because of the high cost of patent litigation. One defense against such misuse is to develop a corresponding portfolio of software patents for defensive purposes. Many software companies, both open source and proprietary, pursue this strategy. In the interests of our company and in an attempt to protect and promote the open source community, Red Hat has elected to adopt this same stance. We do so reluctantly because of the perceived inconsistency with our stance against software patents; however, prudence dictates this position'. See http://www.redhat.com/legal/patent_policy.html.

development of open source code. Legal security can actually be considered as a particular qualitative element of open source software. The problem for OSS communities is thus to produce collective security as well as they produce collective code. It is particularly acute for OSS firms that wish to foster the development and diffusion of OSS, and have therefore an incentive to invest in collective legal security. A large number of examples show that they manage to do so at two different levels, namely by enhancing good practices to cope with patents within the communities, and by developing collective strategies to protect the community against patent threats originating from outside.

6.1. **Open source patents.** Many of the patents that could threaten open source software communities are actually owned by members of these communities. It is usual for large hardware and software firms involved in open source projects to file patents systematically, even though they intend to share the patented programs as open source software. They do so to prevent other firms from doing it in their place and in order to accumulate bargaining chips in case of litigation (Cohen et al., 2000). A first way to reduce legal uncertainty for OSS communities is therefore to ensure that community members will not turn their patents against each other in the future.[25]

So far various initiatives have been taken to turn patents into open source IPRs which use is guaranteed for all members of the community. While some of these initiatives are mere unilateral commitments made by patent owners to encourage the development of open source projects, others involve more stringent collective rules that tend to apply to patents the type of open source licensing requirements that are usually applied to copyright.

A first type of initiative consists for patent holders to pledge unilaterally not to assert their patents against users of a given open source software. This type of commitment is similar to alliances between firms that sell proprietary software, except that they involve a firm and an OSS community. It is illustrated by the agreement reached in 2006 between Microsoft and Novell, a firm specialized in the distribution of Linux. Microsoft pledged not to assert its patents against the version of Linux that is distributed by Novell, thereby facilitating its diffusion amongst risk adverse users. A more general way to freeze his patent rights in order to shelter users and developers against patent infringement consists in committing not to sue those who rely on and adhere to a statement of permitted use. This is exactly what IBM did in announcing in January 2005 that it will release 500 of its software patents into a patent commons available for the open source community.

Some initiatives aim to coordinate and encourage unilateral commitments. Since IBM announced it would release 500 of its patents, several companies have made such patent pledges and covenants. The Open Source Development Lab (hereafter OSDL), a non profit institution financed by large companies and dedicated to the promotion of Linux amongst companies now hosts their patents in a "patent commons" that keeps growing.[26] It provides a central location where patents pledges and software patents (issued and pending) are housed. Commitments not to assert patent rights may concern specific patents (e.g., the 500 IBM US patents) or not.

[25] In that sense, open source communities face the same problem as standard setting organizations.

[26] http://www.patentcommons.org/

Nokia, another contributor to the OSDL patent commons, has for instance committed not to assert all its patents against the Linux kernel. More than ten companies are already listed, including Microsoft that made a patent covenant covering the standard Office 2003 XML reference schemas.[27] Of course, a software patent owner cannot benefit himself from the shield offered by these commitments if she sues a beneficiary of the patent commons for patent infringement. Nokia's patent pledge, for instance, states that "non-assertion commitments are subject to the condition that the party relying on any such commitment and its Affiliates do not assert any of their patents, or patents they control or have a third party assert any patent, against any Linux Kernel." The contributors to the patent commons signal they are ready to counter-attack patent litigation. Patent commons therefore serves two purposes. Firstly, it gives shelter to OSS developers to innovate in using the code in the commons. Secondly, it reduces patent litigation in amassing a defensive patent portfolio that benefits to OSS developers. The patent commons project even encourages the patenting of ideas and then pledging the patents to expand the commons.[28] Initiatives such as the patent commons are more structural than unilateral commitments. However, it is important to notice that participation in such initiatives remains optional for OSS users. Therefore they do not constitute a perfect protection against opportunistic patent holders.

A more stringent possibility consists in incorporating rules concerning patents directly in the open source licences. While the other initiatives are taken unilaterally or multilaterally by firms participating in open source project, licensing clauses are systematic and compulsory for users of the OSS. Patent defence clauses are already explicitly included in recent OSS licenses such as MPL, CPL Apple and Nokia open source licenses. The future issuance of GPLv3 is the major current initiative of this type. As far as software patents are concerned, the inclusion of an express patent license and the introduction of a patent defence clause are the main envisaged changes with respect to the current GPL license.[29] These clauses trigger the termination of the license in case the licensee brings a patent infringement lawsuit against the licensor. As a result, if the licensee sues the licensor, he will have to stop to run or modify the licensed program.[30]

Note that the scope of patent defence clauses varies. They may be strictly limited to lawsuits concerning the licensed software, or may encompass patent attacks to

[27] http://www.patentcommons.org/commons/pledgesearch.php?displaypledge=55

[28] 'The Project also provides a meaningful way for those who oppose software patents to use the current patent system for the benefit of the open source community and industry. Patenting ideas reduces the likelihood that detractors of open source software and open standards will obtain a patent on that same invention and use it against the community and industry, or extract royalties for its use. More importantly, patenting ideas and then pledging the patents in support of The Commons expands and reinforces the protective environment of The Commons'. See http://www.patentcommons.org/about/the_commons.php

[29] The current GPL version, GPLv2, contains a termination clause but it is very narrow. It is triggered when the program infringes a patent, not every time the licensor is sued for patent infringement; furthermore, the termination clause ends the licensee right to distribute not his rights to run and modify the program. The future version of GPL will undoubtedly be more effective.

[30] This retaliation mechanism can only be effective if there is an interest for the licensee to continue to use the programme. More precisely, the licensee has to balance his expected benefits from the patent dispute with her termination costs, that is, the switching costs to another software.

other software of the licensor as well as lawsuits against his customers and users.[31] The larger the scope, the more effective the retaliation mechanism. However too constraining patent clauses may also be rejected by open source users. As for patent commons, their willingness to accept strong commitment must be taken into consideration when designing the license. Because of this participation constraint, there is no guarantee that unilateral or multilateral commitments to freeze patent threats against OSS can entirely suppress the uncertainty generated by software patents. This would require indeed that patent owners accept to give up all the rights their patents confer them, which seems very unlikely.

6.2. **Defence initiatives.** No statistical evidence on changes in individual behaviour of developers before and after SCO is available. However, it seems that many developers have adopted specific defence strategies against patent risk. Those strategies aim to produce effective answers to patent litigation, thereby reducing patent owners' incentives to initiate litigations. They often have an important collective dimension. Contrary to the commitments discussed supra, they can also be effective against opportunistic patent owners that do not belong to the community.

A first possible reply to infringement allegation consists in re-engineering the section of the program in order to circumvent the patent. This is possible for a single firm. However this circumvention strategy is much more powerful and credible if it is implemented cooperatively by a community of OSS developers. OSS developers' capacity and incentives to support it constitute an effective deterrent for opportunistic patent owners.

Some large OSS firms also deter litigation and fight FUD by committing to ensure some protection to users and developers in case of intellectual property right infringement. HP, for instance, provides indemnification and legal defence to its customers for claims by SCO. Red Hat has created a legal defence fund for open source developers. From its part, OSDL has raised several millions of dollars to be able to defend developers, including Linus Tordvals, subjected to Linux related litigation by the SCO group. OSDL has also raised US $4 million to seed a legal center that provides free services (e.g., litigation support, lawyer training) to open source developers and projects. IBM, Intel, Novell and Sun are contributors to these funds. Such initiatives reduce the negative impact of FUD resulting from SCO lawsuits. They signal to future litigators that individual developers and small OSS organizations can resist to patent litigation for they have a large set of good friends, including some with powerful financial strength.[32]

[31]The current draft version of GPLv3 states in section 2 that the license terminates 'if you bring suit against anyone for patent infringement of any of your essential patent claims [...] for making, using, selling [...] a work based on the program'. As a result, assuming that the GPLv3 is issued and quickly substitutes with GLPv2 , the risk of patent litigation for OSS will dramatically decrease because the OSS use the whole economy is huge today and nearly half OSS is licensed under GPL.

[32]Note that some firms also specialize in supplying services that enable the development of patent proof software. For instance, Blackduck Software, a privately held company founded in 2002, proposes software management compliance solutions. Its software platform, ProtexIP, validates software contents and license compliance. Its clients (e.g., Eclipse Foundation) use it to screen code that comes from third parties. OSRM is another example of companies involved in the growing business of mitigating legal risks related to OSS. OSRM offers a risk assessment of the open source components used by firms as well as insurances against claims of patent infringement. Although such firms may work with OSS users, their business model is very different from the

Another collective litigation deterrence strategy consists in gathering and keeping prior art information that may be used to invalidate patents. The communities of developers can use their collaborative methods to more efficiently and rapidly dig around patents. Collecting prior art also benefits from scale economies. Several centralised projects have been initiated. For example, Grokline[33] a community-based, collaborative research project, was designed to trace the ownership history and survivable legal enforcement rights of UNIX and UNIX-like software code. OSDL has also launched a large Prior Art initiative[34] in collaboration with the US Patents and Trademarks Office. It leverages the collaboration of open source communities to build up an operational database of software prior art. The database will provide an effective tool to invalidate patents in case of litigation, but it will also facilitate the work of patent examiners and help them screen valuable patent applications.

7. Conclusion

Contrary to the common wisdom that open Source Software is opposed to intellectual property, we have shown that copyright law is at the heart of open source licensing models, although it is not used for the usual exclusion purpose. We argue that for this reason precisely, the future development of open source software largely depends on considerations relating to intellectual property. We firstly shed light on the problem of license proliferation that may yield incompatibilities between open source software. We then focus on potential problems raised by software patents. Our conclusion is rather optimistic. As proprietary software, open source software is certainly subject to the risk of patent hold up. However, the threat concerns firms that use OSS much more than individual and non-profit OSS users. Therefore, its main impact is to hinder the diffusion of OSS among firms by generating legal uncertainty. Moreover, legal security is a collective good for the OSS community as well as the software itself. Thus members of OSS communities, and foremost firms, have individual and collective incentives to reduce legal uncertainty. Although the resulting collective initiatives may not be sufficient to eliminate any threat due to patents, they confer OSS a significant advantage over proprietary software.

References

Arrow, Kenneth J. (1962), "Economic Welfare and the Allocation of Resources for Inventions", in *The Rate and Direction of Economic Activity: Economic and Social Factors*, Nelson, Richard R. (ed.), Princeton, University Press.

Burk, D. and M. Lemley (2005), "Designing Optimal Software Patents", in *Intellectual Property Rights in Frontier Industries: Software and Biotechnologies*, Hahn, R. (ed.), Washington DC, AEI-Brookings; pp. 81-108.

Coase, Ronald H. (1960), "The Problem of Social Cost", *Journal of Law and Economics*, 3; 1-44.

Coase, Ronald H. (1937), "The Nature of the Firm", *Economica*, 4; 386-405.

Benkler Y. (2002), "Coase's Penguin, or, Linux and The Nature of the Firm", *The Yale Law Journal*, 112; 369.

incentives of OSS users to create collective legal security. Indeed such firms can work as well with pure proprietary software firms.

[33] http://experts.about.com/e/g/gr/Grokline.htm

[34] http://www.osapa.org/

Farrell J. and C. Shapiro (2007), "How Strong are Weak Patents?", University of California at Berkeley Working Paper N^o CPC05-054.

Federal Trade Commission (2003), *To Promote Innovation: A Proper Balance of Competition and Patent Law and Policy*, available at http://www.ftc.gov/opa/2003/10/cpre port.htm.

Fershtman, C. and N. Gandal (2007), "Open Source Software: Motivation and Restrictive Licensing", forthcoming in *Journal of International Economics and Economic Policy*.

Graham, S.,and D. Mowery (2003), "Intellectual Property Protection in the U.S. Software Industry", in Cohen, W.M. and S.A. Merrill (eds.), *Patents in the Knowledge-Based Economy*, Washington DC, National Academies Press; pp. 219-58.

Hall, B. and R. Ziedonis (2001), "The Patent Paradox Revisited: An Empirical Study of Patenting in the US Semiconductor Industry, 1979-95", *RAND Journal of Economics*, 32; 101-28.

Kennedy, D.M. (2001), "A Primer On Open Source Licensing Legal Issues: Copyright, Copyleft and Copyfuture", *Saint Louis University Public Law Review*

Lemley, M. and C. Shapiro (2005), "Probabilistic Patents", *Journal of Economic Perspectives*, 19; 75

Lemley, M. and C. Shapiro (2006), "Patent Holdup and Royalty Stacking", Stanford Law and Economics Olin Working Paper No. 324.

Lerner J. and J. Tirole (2002), "Some Simple Economics of Open Source", *Journal of Industrial Economics*, 52; 197-234.

Lerner J. and J. Tirole (2005a), "The Scope of Open Source Licensing", *Journal of Law, Economics, and Organization*, 21; 20-56.

Lerner J. and J. Tirole (2005b), "The Economics of Technology Sharing: Open Source and Beyond", *Journal of Economic Perspectives*, 19; 99-120.

Lévêque, F. and Y. Ménière (2004), *Economics of Patents and Copyright*, Berkeley CA, Berkeley Electronic Press.

Perens, Bruce (2006), "The Monster Arrives: Software Patent Lawsuits against Open Source Developers", 30 June 2006 http://technocrat.net/d/2006/6/30/5032.

Peters, D.M. (2006), "Software Patents and OSS: An Overview of Patent Provisions in the First Draft for GPLv3", May 3-5 AIPLA 2006 Spring meeting, Chicago.

Rosen, L. (2004), *Open Source Licensing – Software Freedom and Intellectual Property*, Prentice Hall.

Välimäki, M and V. Oksanen (2002), "Open Source Licensing Models for A Company Developing Mass Market Software", proceedings of LawTech 2002, Cambridge MA, USA.

Von Hippel, E. (2002), "Horizontal Innovation Networks by and for Users", MIT Sloan School of Management, Working Paper n° 4366-02.

Lévêque; CERNA, Center in Industrial Economics, Ecole nationale supérieure des mines de Paris, 60 boulevard Saint Michel, 75272 Paris Cedex 06, FRANCE. Ménière; CORE, Université Catholique de Louvain.